What's happened to the Interface since JDK 8

Before JDK 8, the interface type could only have public abstract methods.

JDK 8 introduced the **`default`** method and public **`static`** methods, and JDK 9 introduced **`private`** methods, both static and non-static.

All of these new method types (on the interface) are concrete methods.

The Interface Extension Method - the default method (as of JDK8)

An extension method is identified by the modifier **`default`**, so it's more commonly known as the default method.

This method is a **concrete** method, meaning it has a code block, and we can add statements to it.

In fact, it has to have a method body, even just an empty set of curly braces.

It's a lot like a method on a superclass, because we can override it.

Adding a default method doesn't break any classes currently implementing the interface.

Overriding a default method

So like overriding a method on a class, you have three choices, when you override a default method on an interface.

- You can choose not to override it at all.
- You can override the method and write code for it, so that the interface method isn't executed.
- Or you can write your own code, and invoke the method on the interface, as part of your implementation.